SYSTEM AND METHOD FOR COMBINING REQUESTS FOR DATA BANDWIDTH BY A DATA PROVIDER FOR TRANSMISSION OF DATA OVER AN ASYNCHRONOUS COMMUNICATION MEDIUM

Inventors: Ajay Chandra V. Gummalla
Dolors Sala

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims priority to the following provisional applications:

U.S. Patent Serial No. 60/182,470, entitled "Intelligent Silence Suppression," filed February 15, 2000, by Gummalla *et al.*, (still pending) (incorporated by reference in its entirety herein).

U.S. Patent Serial No. 60/247,188 (Attorney Docket No. CPH 40892 (BP 1560), entitled "A Local Scheduling Mechanism for Cable Modems," filed November 9, 2000, by Sala *et al.*, (still pending) (incorporated by reference in its entirety herein).

U.S. Patent Serial No. 60/254,415 (Attorney Docket No. CPH 40892 (BP 1560.1), entitled "A Local Scheduling Mechanism for Cable Modems," filed December 8, 2000, by Sala *et al.*, (still pending) (incorporated by reference in its entirety herein).

U.S. Patent Serial No. 60/262,201 (Attorney Docket No. CPH 41359 (BP 1702), entitled "Voice Scheduling Algorithms," filed January 17, 2001, by Sala *et al.*, (still pending) (incorporated by reference in its entirety herein).

U.S. Patent Serial No. 60/262,203 (Attorney Docket No. CPH 41362 (BP 1705), entitled "Concatenation of Requests at CMTS," filed January 17, 2001, by Sala et al., (still pending) (incorporated by reference in its entirety herein).

This application claims priority to the following non-provisional application:

U.S. Patent Serial No. 09/427,792, entitled "System and Method for Multiplexing Data from Multiple Sources," filed October 27, 1999, by Limb et al., (still pending) (incorporated by reference in its entirety herein).

This application is related to the following non-provisional applications, all having the same filing date as the present application:

"Method, System and Computer Program Product for Scheduling Upstream Communications", U.S. Patent Serial No. TBD (Attorney Docket No. 1875.0440002, by Gummalla *et al.* (incorporated by reference in its entirety herein).

"System and Method to Support Constant Bit Rate Services in a Shared Communication System," U.S. Patent Serial No. TBD (Attorney Docket No. 40672/LTR/B600) by Gummalla *et al.* (incorporated by reference in its entirety herein).

"System and Method for Suppressing Silence in Voice Traffic over an Asynchronous Communication Medium ," U.S. Patent Serial No. TBD (Attorney Docket No. 1875.0430001) by Gummalla *et al.*, (incorporated by reference in its entirety herein).

"Cable Modem System and Method for Specialized Data Transfer," U.S. Patent Serial No. TBD (Attorney Docket No. 1875.0460001) by Bunn *et al.*, (incorporated by reference in its entirety herein).

BACKGROUND OF THE INVENTION

Field of the Invention

[0001] The present invention is generally related to increasing the efficiency of providing requested bandwidth to a data provider via asynchronous communication mediums.

Background Art

[0002] The importance to the modern economy of rapid data access and exchange cannot be overstated. This explains the exponentially increasing popularity of the data access and exchange via cable networks (including coaxial cable or Hybrid fiber coaxial cable), the Internet, intranets, wireless networks, satellites and so forth (i.e., communication mediums). Rapid data access and exchange is partly dependent upon how efficiently bandwidth is allocated to a data provider in order for the data provider to transfer the requested data to a user via one of the communication mediums mentioned above.

[0003] One very desirable solution for rapid data access and exchange is via cable networks and cable modems. Cable modems provide asynchronous communications on cable networks. In general, a user connects a cable modem to the TV outlet for his or her cable TV, and the cable TV operator connects a cable modem termination system ("CMTS") in the operator's headend. The CMTS is a central device for connecting the cable network to a data network like the Internet. The CMTS is a central distribution point for a cable system. Data flows "downstream" from the CMTS to the cable modem (i.e., downstream communication). Alternatively, data flows "upstream" from the cable modem to the CMTS (i.e., upstream communication).

[0004] A common cable modem standard today is the Data Over Cable Service Interface Specification ("DOCSIS"). DOCSIS defines technical specifications

for both cable modems and CMTS. DOCSIS downstream communication is quite restrictive in the way the control information is conveyed to the data provider (e.g., cable modem) via a DOCSIS CMTS scheduler. What is needed is to override the CMTS scheduler of DOCSIS and provide a CMTS scheduler that reduces the overhead of bandwidth grants via downstream communication by providing flexibility in the allocation of the bandwidth.

BRIEF SUMMARY OF THE INVENTION

[0005] The present invention is a system and method for combining requests for bandwidth by a data provider for transmission of data over an asynchronous communication medium. An embodiment of the method comprises the first step of receiving bandwidth requests from one or more data providers, each bandwidth request having a data provider identifier, a priority identifier, and the amount of required bandwidth. The next step is storing each of the bandwidth requests in a data structure so as to maintain the order in which the bandwidth requests were received. The third step is, based on the priority identifier and the order of each bandwidth request, scheduling each the bandwidth request in an order to be serviced. The next step is combining each of the bandwidth requests having the same the data provider identifier into a data burst bandwidth. The final step is granting the data burst bandwidth to the data provider over the asynchronous communication medium.

[0006] In another embodiment of the method of the present invention, the first step involves receiving bandwidth requests from one or more data providers, each bandwidth request having a data provider identifier, a priority identifier, and the amount of required bandwidth. The second step is combining, by data provider identifier and priority identifier, the amount of bandwidth required to represent a data burst bandwidth. The fourth step is, based on one or more quality of service parameters (which varies from priority to priority), scheduling each the data burst bandwidth in an order to be serviced. The final step involves granting

the data burst bandwidth to the data provider over the asynchronous communication medium.

[0007] An embodiment of the system comprises a headend and a scheduler that are coupled together. The scheduler receives bandwidth requests from one or more data providers, each bandwidth request having a data provider identifier, a priority identifier, and the amount of required bandwidth. The scheduler then stores each of the bandwidth requests in a data structure so as to maintain the order in which the bandwidth requests were received. Then the scheduler schedules each the bandwidth request in an order to be serviced based on the priority identifier and the order of each bandwidth request. Next, the scheduler combines each of the bandwidth requests having the same the data provider identifier into a data burst bandwidth. Finally, the headend grants the data burst bandwidth to the data provider over the asynchronous communication medium.

[0008] Another embodiment of the system also includes a headend and a scheduler coupled together. Here, the scheduler receives bandwidth requests from one or more data providers, each bandwidth request having a data provider identifier, a priority identifier, and the amount of required bandwidth. The scheduler then combines, by data provider identifier and priority identifier, the amount of bandwidth required to represent a data burst bandwidth. The scheduler, based on one or more quality of service parameters (which varies from priority to priority), schedules each the data burst bandwidth in an order to be serviced. Finally, headend grants the data burst bandwidth to the data provider over the asynchronous communication medium.

BRIEF DESCRIPTION OF THE DRAWINGS/FIGURES

[0009] The present invention will be described with reference to the accompanying drawings, wherein:

[0010] FIG. 1 is a block diagram representing an example operating environment of the present invention according to an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

A. Overview of the Invention

[0020] For illustration purposes, the present invention is described in terms of being utilized with a cable network. It should be understood that the present invention is not limited to use with a cable network. In fact, the present invention may be used with any communication medium, including but not limited to, the Internet, intranets, fiber optic networks, wireless networks and satellites.

[0021] Data in the present invention includes any type of information. This includes, but is not limited to, digital, voice, video, audio, etc.

B. System Architecture Overview

[0022] FIG. 1 is a block diagram representing an example operating environment of the present invention. It should be understood that the example operating environment in FIG. 1 is shown for illustrative purposes only and does not limit the invention. Other implementations of the operating environment described herein will be apparent to persons skilled in the relevant art(s) based on the teachings contained herein, and the invention is directed to such other implementations. Referring to FIG. 1, a CMTS 102, a cable modem 104, downstream communication 106 and upstream communication 108, are shown. CMTS 102 further includes a CMTS scheduler 102 and a data structure 112. Cable modem 104 includes a cable modem scheduler 116. Each of these components will be briefly described next.

[0023] In general, cable modem 104 forwards or provides data via asynchronous communications on cable networks. Cable modem 104 receives data from a user that needs to be transferred via a cable network. In order to do this, cable modem 104 requests that CMTS 102 grant to it the necessary bandwidth.

[0024] As mentioned, cable modem 104 receives data from a user to be transferred via a cable network. Different types of data require different modes of transfer since the importance of timing is different with different types of data. For example, voice data cannot tolerate delays in its transfer. Alternatively, the type of data involved in web surfing can tolerate delays in its transfer.

[0025] In order to ensure the importance of timing is maintained, cable modem 104 assigns different priority identifiers to different types of data. The higher the priority a data has, the less of a delay that type of data will experience in its transfer via the cable network. Thus, voice data would be assigned a priority identifier with a higher priority than data involved in web surfing.

[0026] Cable modem scheduler 116 is coupled to cable modem 104. Cable modem scheduler 116 is described in detail in related non-provisional application, entitled "Method, System and Computer Program Product for Scheduling Upstream Communications", (Attorney Docket No. 1875.0440002), filed concurrently with the present invention (incorporated herein by reference in its entirety). In general, cable modem scheduler 116 is responsible for multiplexing the internal traffic, (i.e., requesting the necessary bandwidth that cable modem 116 needs to transfer its current types of data). Cable modem scheduler 116 must take into consideration the different priorities given to the current data to be transferred and to request bandwidth from CMTS 102 accordingly.

[0027] As stated above, DOCSIS is a common cable modem standard used today. DOCSIS provides standard centralized scheduling decisions that do not allow for flexibility in terms of deciding when cable modem 104 requests bandwidth from CMTS 102 in order to transfer its current data. Cable modem scheduler 116 defines an architecture that overrules this DOCSIS standard in a seamless manner. Some of the main differences between the DOCSIS standard and cable modem scheduler 116 are described next.

[0028] One main difference between the DOCSIS standard and cable modem scheduler 116 includes the decoupling of the request phase with the grant phase (i.e., grants of bandwidth received from CMTS 102). Here, cable modem

[0031] CMTS 102 is a central device for connecting the cable network to a data network. CMTS scheduler 102 is a bandwidth manager. CMTS scheduler 102, as a bandwidth manager, decides how to grant available bandwidth according to the current bandwidth requests. This grant is done via downstream communication 106 in such as way as to reduce overhead. This ability to decide how to grant available bandwidth provides flexibility. This flexibility allows the present invention to reduce the overhead involved in granting bandwidth to cable modem 104 via downstream communication. Data structure 112 is used to organized the received bandwidth requests in such as way as to take into consideration the type of data (via the priority identifiers) and the order in which the requests were received.

[0032] As described above, the way in which the DOCSIS CMTS scheduler grants bandwidth is quite restrictive, thus creating unnecessary overhead in downstream communication. The details of how CMTS scheduler 110 grants bandwidth to cable modem 104 so that to decrease overhead will be described in detail below. The overhead in each granted bandwidth will be described next with reference to FIG. 2.

C. Granted Bandwidth Overhead

[0033] FIG. 2 illustrates an example of granted bandwidth 202. As shown, granted bandwidth shows overhead 204 including a preamble, a guard band and a forward error correction ("FEC"). The preamble is a pattern of bits transmitted at the start of a frame used to implement transmitter and receiver synchronization. The guard band is the time left vacant between adjacent transmissions to allow for detection certainty and clock synchronization inaccuracies in an asynchronous communication system. The FEC is the process whereby additional bits are appended to a block of bits so that the receiver will be able to both detect and correct transmission errors.

[0034] Overhead 204 may be referred to as the physical layer overhead. Regardless of the size of granted bandwidth 202, overhead 204 uses approximately the same amount of bytes. Where the preamble and guard band are typically fixed in size, the FEC is a variable size that depends on the amount of bandwidth. Therefore, the larger the granted bandwidth, the greater the efficiency. However, larger granted bandwidths mean greater end-to-end delay experienced by a voice call, for example. Thus, when granting bandwidth, end-to-end delay versus gained efficiency must be taken into consideration.

[0035] Different systems may reserve different amount of bytes for overhead 204. How the present invention combines bandwidth requests from the same cable modem 104 to create a single data burst bandwidth is described next.

D. Combination of Bandwidth Grants

[0036] As mentioned above, cable modem scheduler 116 may send different bandwidth requests to CMTS 102, including but not limited to, voice, piggyback and data for activities such as web surfing. Thus, at any given time, CMTS 102 may have more than one request for the same cable modem. This is especially true since cable modem scheduler 104 piggybacks requests as often as possible. These requests may have different priority identifiers values or the same priority identifier value for data that has arrived at CMTS 102 at different times.

[0037] A goal of the present invention is to combine all pending bandwidth requests from the same cable modem into one data burst bandwidth request. Here, instead of individual requests being granted which would require physical overhead for each grant, the present invention reduces the physical over to one for all of the individual requests by combining them. This can be accomplished by the present invention partly because there is a decoupling of the request phase (i.e., bandwidth requests from cable modem 104) with the grant phase (i.e., grants of bandwidth received from CMTS 102), as mentioned above. CMTS scheduler 110 needs to take into consideration end-to-end delay verses the gained

efficiency. FIG. 3 is a high level flowchart that describes the process of combining bandwidth requests according to an embodiment of the present invention.

[0038] In FIG. 3, control starts at step 302. In step 302, CMTS 102 receives one or more bandwidths requests from one or more cable modems 104 via upstream communication 108. Control then passes to step 304.

[0039] In step 304, CMTS scheduler 110 combines one or more bandwidths requests from the same cable modem 104 to create a single data burst bandwidth. Control then passes to step 306.

[0040] In step 306, CMTS 102 grants the data burst bandwidth to the appropriate cable modem 104 via downstream communication 106. The flowchart in FIG. 3 ends at this point. Next, two different embodiments of step 304 are described. The first embodiment is described with reference to FIGs. 4-6. Here, bandwidth requests are scheduled to be serviced based on priority identifiers (i.e., type of data) and the order in which the requests for bandwidth were received. The second embodiment of step 304 is described with reference to FIGs. 8 and 9. Here, bandwidth requests are scheduled to be serviced based on various quality of service parameters.

[0041] FIG. 4 describes in detail the step of combining bandwidth requests from the same data provider (e.g., cable modem 104) to create a data burst bandwidth (step 304 of FIG. 3) according to an embodiment of the present invention. In FIG. 4, control starts at step 402. In step 402, as CMTS 102 receives bandwidth requests from one or more cable modems 104, CMTS scheduler 112 stores each bandwidth request in data structure 112. In an embodiment of the present invention, data structure 112 is implemented as one or more queues and CMTS scheduler 112 is implements as a priority first come first serve scheduler. Data structure 112 as implemented as one or more queues is shown in FIG. 5.

[0042] FIG. 5 illustrates multiple queues with priority identifiers values 1 through n. Queue 502 stores the data request with the highest priority (i.e., 0). Queue 508 stores the data with the lowest priority (i.e., n). Queues 504 and 506

store data with priority values in between the highest and the lowest. In any event, each queue has its own priority value and all requests stored in any of the queues must have the same priority identifier values.

[0043] As illustrated in FIG. 5, queue 502 currently stores three bandwidth requests, including requests 510-514. Queue 504 stores two bandwidth request, including requests 516 and 518. Neither queue 506 or queue 508 are currently storing any bandwidth request.

[0044] Each bandwidth request is indicated as (n,m), where n is the data provider identifier and m is the amount of required bandwidth. As described above, each bandwidth request includes at least three fields, including the data provider identifier, the priority identifier, and the amount of required bandwidth. For example, bandwidth request 510 has a priority identifier value of 0, the cable modem 104 who sent the request has the data provider identifier of 1 and it is requesting 5 units of bandwidth. In addition, bandwidth request 510 arrived at CMTS 102 before bandwidth request 512, which in turn arrived before bandwidth request 514, as indicated by their positions in queue 502. It is important to note that bandwidth request 516 may arrive at CMTS 102 before bandwidth request 514, but bandwidth request 514 is still likely to be granted bandwidth first due to it having a higher priority identifier.

[0045] Also illustrated by FIG. 5 is that the cable modem 104 that has the data provider value of 2 has two bandwidth requests, including requests 512 and 516. It is important to note that these two requests are for data that have different priority identifier values (e.g., requests for voice and piggyback). Control then passes to step 404.

[0046] In step 404, CMTS scheduler 110 schedules each bandwidth request stored in data structure 112 in an order to be serviced by the present invention. Here, bandwidth requests are scheduled to be serviced based on priority identifiers (i.e., type of data) and the order in which the requests for bandwidth were received. An example scheduling that is based on the example data structure shown in FIG. 5 is shown in FIG. 6 as schedule 602. In schedule 602,

request 510 is scheduled first, request 512 is scheduled second, and so forth. Control then passes to step 406.

[0047] In step 406, CMTS scheduler 110 utilizes schedule 602 to combine bandwidth requests from the same cable modem 104 to create a single data burst bandwidth for each cable modem 104 (i.e., data provider identifier). This is illustrated in FIG. 7. In FIG. 7, data burst bandwidth 702 is created by combining bandwidth requests 510 and 514. Data burst bandwidth 704 is created by combining bandwidth requests 512 and 516. Finally, data burst bandwidth 706 is identical to bandwidth request 518. It is important to note that the conversion between storage units may alter the units of storage required. For example, this account for why data burst bandwidth 702 is (1,6) and not (1,7). This is for illustration purposes only and is not meant to limit the present invention.

[0048] At this point, data burst bandwidths 702-706 are ready to be granted to their appropriate cable modems as accomplished in step 306 (FIG. 3). The flowchart in FIG. 4 ends at this point. The second embodiment of step 304 will now be described with reference to FIGs. 8 and 9.

[0049] As mentioned above, bandwidth requests in this embodiment are scheduled to be serviced based on various quality of service parameters.

[0050] FIG. 8 describes in detail the step of combining bandwidth requests from the same data provider (e.g., cable modem 104) to create a data burst bandwidth (step 304 of FIG. 3) according to another embodiment of the present invention. In FIG. 8, control starts at step 802. In step 802, as CMTS 102 receives bandwidth requests from one or more cable modems 104, CMTS scheduler 112 combines or accumulates the sum of the units of bandwidth requested per priority identifier and data provider identifier. This sum represents what will be required to represent the data burst bandwidths. The accumulation of units of bandwidth requested is illustrated in FIG. 9.

[0051] In FIG. 9, a table 902 is shown. The columns of table 902 represent the priority identifier (i.e., type of data in the request). The rows of table 902 represent the data provider identifiers. The cells of table 902 represent the

[0031] CMTS 102 is a central device for connecting the cable network to a data network. CMTS scheduler 102 is a bandwidth manager. CMTS scheduler 102, as a bandwidth manager, decides how to grant available bandwidth according to the current bandwidth requests. This grant is done via downstream communication 106 in such as way as to reduce overhead. This ability to decide how to grant available bandwidth provides flexibility. This flexibility allows the present invention to reduce the overhead involved in granting bandwidth to cable modem 104 via downstream communication. Data structure 112 is used to organized the received bandwidth requests in such as way as to take into consideration the type of data (via the priority identifiers) and the order in which the requests were received.

[0032] As described above, the way in which the DOCSIS CMTS scheduler grants bandwidth is quite restrictive, thus creating unnecessary overhead in downstream communication. The details of how CMTS scheduler 110 grants bandwidth to cable modem 104 so that to decrease overhead will be described in detail below. The overhead in each granted bandwidth will be described next with reference to FIG. 2.

C. Granted Bandwidth Overhead

[0033] FIG. 2 illustrates an example of granted bandwidth 202. As shown, granted bandwidth shows overhead 204 including a preamble, a guard band and a forward error correction ("FEC"). The preamble is a pattern of bits transmitted at the start of a frame used to implement transmitter and receiver synchronization. The guard band is the time left vacant between adjacent transmissions to allow for detection certainty and clock synchronization inaccuracies in an asynchronous communication system. The FEC is the process whereby additional bits are appended to a block of bits so that the receiver will be able to both detect and correct transmission errors.

[0034] Overhead 204 may be referred to as the physical layer overhead. Regardless of the size of granted bandwidth 202, overhead 204 uses approximately the same amount of bytes. Where the preamble and guard band are typically fixed in size, the FEC is a variable size that depends on the amount of bandwidth. Therefore, the larger the granted bandwidth, the greater the efficiency. However, larger granted bandwidths mean greater end-to-end delay experienced by a voice call, for example. Thus, when granting bandwidth, end-to-end delay versus gained efficiency must be taken into consideration.

[0035] Different systems may reserve different amount of bytes for overhead 204. How the present invention combines bandwidth requests from the same cable modem 104 to create a single data burst bandwidth is described next.

D. Combination of Bandwidth Grants

[0036] As mentioned above, cable modem scheduler 116 may send different bandwidth requests to CMTS 102, including but not limited to, voice, piggyback and data for activities such as web surfing. Thus, at any given time, CMTS 102 may have more than one request for the same cable modem. This is especially true since cable modem scheduler 104 piggybacks requests as often as possible. These requests may have different priority identifiers values or the same priority identifier value for data that has arrived at CMTS 102 at different times.

[0037] A goal of the present invention is to combine all pending bandwidth requests from the same cable modem into one data burst bandwidth request. Here, instead of individual requests being granted which would require physical overhead for each grant, the present invention reduces the physical over to one for all of the individual requests by combining them. This can be accomplished by the present invention partly because there is a decoupling of the request phase (i.e., bandwidth requests from cable modem 104) with the grant phase (i.e., grants of bandwidth received from CMTS 102), as mentioned above. CMTS scheduler 110 needs to take into consideration end-to-end delay verses the gained

efficiency. FIG. 3 is a high level flowchart that describes the process of combining bandwidth requests according to an embodiment of the present invention.

[0038] In FIG. 3, control starts at step 302. In step 302, CMTS 102 receives one or more bandwidths requests from one or more cable modems 104 via upstream communication 108. Control then passes to step 304.

[0039] In step 304, CMTS scheduler 110 combines one or more bandwidths requests from the same cable modem 104 to create a single data burst bandwidth. Control then passes to step 306.

[0040] In step 306, CMTS 102 grants the data burst bandwidth to the appropriate cable modem 104 via downstream communication 106. The flowchart in FIG. 3 ends at this point. Next, two different embodiments of step 304 are described. The first embodiment is described with reference to FIGs. 4-6. Here, bandwidth requests are scheduled to be serviced based on priority identifiers (i.e., type of data) and the order in which the requests for bandwidth were received. The second embodiment of step 304 is described with reference to FIGs. 8 and 9. Here, bandwidth requests are scheduled to be serviced based on various quality of service parameters.

[0041] FIG. 4 describes in detail the step of combining bandwidth requests from the same data provider (e.g., cable modem 104) to create a data burst bandwidth (step 304 of FIG. 3) according to an embodiment of the present invention. In FIG. 4, control starts at step 402. In step 402, as CMTS 102 receives bandwidth requests from one or more cable modems 104, CMTS scheduler 112 stores each bandwidth request in data structure 112. In an embodiment of the present invention, data structure 112 is implemented as one or more queues and CMTS scheduler 112 is implements as a priority first come first serve scheduler. Data structure 112 as implemented as one or more queues is shown in FIG. 5.

[0042] FIG. 5 illustrates multiple queues with priority identifiers values 1 through n. Queue 502 stores the data request with the highest priority (i.e., 0). Queue 508 stores the data with the lowest priority (i.e., n). Queues 504 and 506

store data with priority values in between the highest and the lowest. In any event, each queue has its own priority value and all requests stored in any of the queues must have the same priority identifier values.

[0043] As illustrated in FIG. 5, queue 502 currently stores three bandwidth requests, including requests 510-514. Queue 504 stores two bandwidth request, including requests 516 and 518. Neither queue 506 or queue 508 are currently storing any bandwidth request.

[0044] Each bandwidth request is indicated as (n,m), where n is the data provider identifier and m is the amount of required bandwidth. As described above, each bandwidth request includes at least three fields, including the data provider identifier, the priority identifier, and the amount of required bandwidth. For example, bandwidth request 510 has a priority identifier value of 0, the cable modem 104 who sent the request has the data provider identifier of 1 and it is requesting 5 units of bandwidth. In addition, bandwidth request 510 arrived at CMTS 102 before bandwidth request 512, which in turn arrived before bandwidth request 514, as indicated by their positions in queue 502. It is important to note that bandwidth request 516 may arrive at CMTS 102 before bandwidth request 514, but bandwidth request 514 is still likely to be granted bandwidth first due to it having a higher priority identifier.

[0045] Also illustrated by FIG. 5 is that the cable modem 104 that has the data provider value of 2 has two bandwidth requests, including requests 512 and 516. It is important to note that these two requests are for data that have different priority identifier values (e.g., requests for voice and piggyback). Control then passes to step 404.

[0046] In step 404, CMTS scheduler 110 schedules each bandwidth request stored in data structure 112 in an order to be serviced by the present invention. Here, bandwidth requests are scheduled to be serviced based on priority identifiers (i.e., type of data) and the order in which the requests for bandwidth were received. An example scheduling that is based on the example data structure shown in FIG. 5 is shown in FIG. 6 as schedule 602. In schedule 602,

request 510 is scheduled first, request 512 is scheduled second, and so forth. Control then passes to step 406.

[0047] In step 406, CMTS scheduler 110 utilizes schedule 602 to combine bandwidth requests from the same cable modem 104 to create a single data burst bandwidth for each cable modem 104 (i.e., data provider identifier). This is illustrated in FIG. 7. In FIG. 7, data burst bandwidth 702 is created by combining bandwidth requests 510 and 514. Data burst bandwidth 704 is created by combining bandwidth requests 512 and 516. Finally, data burst bandwidth 706 is identical to bandwidth request 518. It is important to note that the conversion between storage units may alter the units of storage required. For example, this account for why data burst bandwidth 702 is (1,6) and not (1,7). This is for illustration purposes only and is not meant to limit the present invention.

[0048] At this point, data burst bandwidths 702-706 are ready to be granted to their appropriate cable modems as accomplished in step 306 (FIG. 3). The flowchart in FIG. 4 ends at this point. The second embodiment of step 304 will now be described with reference to FIGs. 8 and 9.

[0049] As mentioned above, bandwidth requests in this embodiment are scheduled to be serviced based on various quality of service parameters.

[0050] FIG. 8 describes in detail the step of combining bandwidth requests from the same data provider (e.g., cable modem 104) to create a data burst bandwidth (step 304 of FIG. 3) according to another embodiment of the present invention. In FIG. 8, control starts at step 802. In step 802, as CMTS 102 receives bandwidth requests from one or more cable modems 104, CMTS scheduler 112 combines or accumulates the sum of the units of bandwidth requested per priority identifier and data provider identifier. This sum represents what will be required to represent the data burst bandwidths. The accumulation of units of bandwidth requested is illustrated in FIG. 9.

[0051] In FIG. 9, a table 902 is shown. The columns of table 902 represent the priority identifier (i.e., type of data in the request). The rows of table 902 represent the data provider identifiers. The cells of table 902 represent the

accumulation of units of requested bandwidth by priority identifier and data provider identifier. For illustration purposes, it is assumed that the same bandwidth requests are received by CMTS 102 as received and explained with reference to FIG. 5 above. Therefore, an accumulation 904 (with the value 7) is calculated by bandwidth request 510 and bandwidth request 514 (requesting 5 and 2 units of bandwidth, respectively). Control then passes to step 804.

[0052] In step 804, CMTS scheduler 110 schedules each data burst bandwidth based on quality of service parameters. Quality of service parameters include efficiency of transmission and transfer delay tolerance. As stated above, different types of data require different modes of transfer since the importance of timing is different with different types of data. For example, voice data cannot tolerate delays in its transfer. Alternatively, the type of data involved in web surfing can tolerate delays in its transfer.

[0053] The priority identifier specified in the request is not significantly important since the cable modem scheduler does not use it 104 when transferring data via the bandwidth. For consistency in all cases is desired, the present invention may specify that the priority identifier of the data burst bandwidth gets the lowest priority that contributes to the accumulated sum. Also note that this embodiment may choice to incorporate scheduling, as shown in FIG. 6 above, and the combination of bandwidth requests, as shown in FIG. 7 above. At this point, data burst bandwidths are ready to be granted to their appropriate cable modems as accomplished in step 306 (FIG. 3). The flowchart in FIG. 8 ends at this point.

E. Example Environment of the Present Invention

[0054] CMTS 102, CMTS scheduler 110 and cable modem scheduler 116 may be implemented using computer 1000 as shown in FIG. 10. Obviously, more than one of these functional components could be implemented on a single computer 1000.

[0055] The present invention may be implemented using hardware, software or a combination thereof and may be implemented in a computer system or other processing system. In fact, in one embodiment, the invention is directed toward one or more computer systems capable of carrying out the functionality described herein. The computer system 1000 includes one or more processors, such as processor 1004. The processor 1004 is connected to a communication bus 1006. Various software embodiments are described in terms of this example computer system. After reading this description, it will become apparent to a person skilled in the relevant art how to implement the invention using other computer systems and/or computer architectures.

[0056] Computer system 1000 also includes a main memory 1008, preferably random access memory (RAM), and can also include a secondary memory 1010. The secondary memory 1010 can include, for example, a hard disk drive 1012 and/or a removable storage drive 1014, representing a floppy disk drive, a magnetic tape drive, an optical disk drive, etc. The removable storage drive 1014 reads from and/or writes to a removable storage unit 1018 in a well known manner. Removable storage unit 1018, represents a floppy disk, magnetic tape, optical disk, etc. which is read by and written to by removable storage drive 1014. As will be appreciated, the removable storage unit 1018 includes a computer usable storage medium having stored therein computer software and/or data.

[0057] In alternative embodiments, secondary memory 1010 may include other similar means for allowing computer programs or other instructions to be loaded into computer system 1000. Such means can include, for example, a removable storage unit 1022 and an interface 1020. Examples of such can include a program cartridge and cartridge interface (such as that found in video game devices), a removable memory chip (such as an EPROM, or PROM) and associated socket, and other removable storage units 1022 and interfaces 1020 which allow software and data to be transferred from the removable storage unit 1018 to computer system 1000.

[0058] Computer system 1000 can also include a communications interface 1024. Communications interface 1024 allows software and data to be transferred between computer system 1000 and external devices. Examples of communications interface 1024 can include a modem, a network interface (such as an Ethernet card), a communications port, a PCMCIA slot and card, etc. Software and data transferred via communications interface 1024 are in the form of signals which can be electronic, electromagnetic, optical or other signals capable of being received by communications interface 1024. These signals 1026 are provided to communications interface via a channel 1028. This channel 1028 carries signals 1026 and can be implemented using wire or cable, fiber optics, a phone line, a cellular phone link, an RF link and other communications channels.

[0059] In this document, the terms "computer program medium" and "computer usable medium" are used to generally refer to media such as removable storage device 1018, a hard disk installed in hard disk drive 1012, and signals 1026. These computer program products are means for providing software to computer system 1000.

[0060] Computer programs (also called computer control logic) are stored in main memory and/or secondary memory 1010. Computer programs can also be received via communications interface 1024. Such computer programs, when executed, enable the computer system 1000 to perform the features of the present invention as discussed herein. In particular, the computer programs, when executed, enable the processor 1004 to perform the features of the present invention. Accordingly, such computer programs represent controllers of the computer system 1000.

[0061] In an embodiment where the invention is implemented using software, the software may be stored in a computer program product and loaded into computer system 1000 using removable storage drive 1014, hard drive 1012 or communications interface 1024. The control logic (software), when executed by the processor 1004, causes the processor 1004 to perform the functions of the invention as described herein.

[0062] In another embodiment, the invention is implemented primarily in hardware using, for example, hardware components such as application specific integrated circuits (ASICs). Implementation of the hardware state machine so as to perform the functions described herein will be apparent to persons skilled in the relevant art(s). In yet another embodiment, the invention is implemented using a combination of both hardware and software.

F. Conclusion

[0063] While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example, and not limitation. It will be apparent to persons skilled in the relevant art that various changes in form and detail may be made therein without departing from the spirit and scope of the invention. This is especially true in light of technology and terms within the relevant art(s) that may be later developed. Thus, the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.